# Improved approximations for robust mincut and shortest path

Valentin Polishchuk Mikko Sysikaski
Helsinki Institute for Information Technology
firstname.lastname@cs.helsinki.fi

**Abstract**

In two-stage robust optimization the solution to a problem is built in two stages: In the first stage a partial, not necessarily feasible, solution is exhibited. Then the adversary chooses the "worst" scenario from a predefined set of scenarios. In the second stage, the first-stage solution is extended to become feasible for the chosen scenario. The costs at the second stage are larger than at the first one, and the objective is to minimize the total cost paid in the two stages.

We give a 2-approximation algorithm for the robust mincut problem and a $(\gamma+2)$-approximation for the robust shortest path problem, where $\gamma$ is the approximation ratio for the Steiner tree. This improves the factors $1+\sqrt{2}$ and $2(\gamma+2)$ from [Golovin, Goyal and Ravi. Pay today for a rainy day: Improved approximation algorithms for demand-robust min-cut and shortest path problems. *STACS 2006*]. In addition, our solution for robust shortest path is simpler and more efficient than the earlier ones; this is achieved by a more direct algorithm and analysis, not using some of the standard demand-robust optimization techniques.

**Keywords:** Approximation algorithms, Demand-robust optimization

## 1 Introduction

The general setting in a two-stage optimization problem is as follows: There is a set of *demands* (aka *scenarios*), one of which has to be satisfied tomorrow. It is not until tomorrow that it is revealed which demand must be satisfied. A demand is satisfied by buying a set of *resources*. Thus, one possibility to satisfy the tomorrow's demand is to wait until tomorrow, know the scenario, and buy a corresponding set of resources. However, the resources are cheaper today than tomorrow, by an "inflation" factor $\lambda > 1$. Hence it makes sense to buy some resources already today, i.e., at the *first stage*, even without knowing the tomorrow's scenario. (Say, if $\lambda = \infty$, the resources bought today should better satisfy all demands.) Then tomorrow, upon revealing the requested demand, only some additional, *second-stage*, resources have to be bought at the higher price.

The tomorrow's demand is chosen by an adversary. The adversary knows the resources bought at the first stage. He also knows the algorithm that you will use for buying second-stage resources. The adversary chooses the scenario so that your second-stage cost is as large possible (the adversary is omnipotent, and can solve an NP-hard problem for that, if necessary). Your objective is thus to minimize the *maximum, worst-case* cost paid in the two stages. Because of such hedging against the worst demand, this type of two-stage robust optimization is called *demand-robust*.

## Related work

In *stochastic optimization* (see, e.g., [8, 9]) the objective is to minimize the *expected* cost paid over the two stages. *Universal approximations* [5, 10] may in a sense be viewed as *one-stage* robust solutions. The *demand-robust optimization* as studied in this paper was introduced by Dhamdhere et al. in [2]. Several techniques have proved to be viable in the field:

*Minimal feasible solutions.* Dhamdhere et al. [2] showed that there always exists an approximate first-stage solution which is a minimal feasible solution for a subset of scenarios. Restricting oneself to such solutions makes one loose at most a factor of 2 in comparison with the unrestricted case. Since the pioneering paper [2], the minimal-solution idea has been extensively used in the design of approximation algorithms for two-stage robust optimization problems.

*LP rounding.* IP formulations of optimization problems often extend directly to stochastic and demand-robust versions; rounding the LP relaxation solution is one way to obtain an approximation.

*Thresholded $\alpha$-approximations.* A common approach to demand-robust optimization is as follows: Suppose you are shooting for an $\alpha$-approximation. Guess the second-stage cost $C^*_{II}$ of the optimal solution (often the number of relevant $C^*_{II}$s is small; if worse comes to worst, go through "all possible" $C^*_{II}$s approximately with repeated doubling – or more precisely, with repeated $(1+\varepsilon)$-ing). In the first stage, satisfy all high-cost demands – those each of which is more expensive than $\alpha C^*_{II}$ to satisfy. Then in the second stage you are guaranteed to pay at most $\alpha C^*_{II}$ – which is within factor $\alpha$ of optimal second-stage cost. Finally, argue that your first-stage solution is also within $\alpha$ times the first-stage cost of the optimum – for the overall approximation guarantee of $\alpha$. A very general treatment of the thresholded covering algorithms is presented in a recent paper [7].

## Our contributions

In Section 2 we present a thresholded 2-approximation for the robust mincut problem. This improves the (also thresholded) $(1+\sqrt{2})$-approximation from [4]. The improved approximation guarantee is due to a refined analysis using,

similarly to [4], laminarity of mincuts (the Gomory-Hu mincuts tree).

In Section 3 we give a $(\gamma + 2)$-approximation algorithm for the robust shortest path problem, where $\gamma$ is the Steiner tree approximation ratio. This improves the $2(\gamma + 2)$-approximation from [4] (the techniques in [6] potentially imply a 4.25-approximation). The algorithm and its analysis are very simple. In particular, unlike [4] we do not restrict ourselves to minimal feasible solutions and do not use the thresholding. Avoiding the guessing of the second-stage cost of the optimum makes our algorithm more efficient (by at least a linear factor) than that of [4].

## 2 Demand-robust mincut

In the demand-robust mincut problem the input is a (positively) weighted undirected graph $G = (V, E, C)$ with $C$ representing the *capacities* of edges, the *root* vertex $r \in V$, and a set $T \subseteq V \setminus r$ of *terminals*. For a terminal $t \in T$ let $\mathsf{m}(t)$ denote the minimum $r$-$t$ cut (if the mincut is not unique, take $\mathsf{m}(t)$ to be the cut that cuts out from $r$ a maximal set of vertices); similarly, for a set $S \subseteq T$ of terminals, $\mathsf{m}(S)$ is the minimum $r$-$S$ cut. We use $C(t), C(S)$ to denote the capacities $C(\mathsf{m}(t)), C(\mathsf{m}(S))$ of the mincuts $\mathsf{m}(t), \mathsf{m}(S)$. For a subset $E' \subseteq E$ of edges let $\mathsf{m}_{E'}(t)$ be the minimum $r$-$t$ cut in $G$ with weights of edges in $E'$ set to 0; let $C_{E'}(t)$ denote the capacity of the mincut $\mathsf{m}_{E'}(t)$.

A feasible solution to the robust mincut problem is an arbitrary set $E_I \subseteq E$ of edges. The cost of the solution is

$$C(E_I) + \lambda \cdot \max_{t \in T} C_{E_I}(t)$$

where $\lambda$ is the inflation factor.

The edges $E_I$ of the solution are called the *first-stage* edges and the cost $C(E_I)$ — *first-stage* cost; the edges $\mathsf{m}_{E_I}(t)$ are the *second-stage* edges for terminal $t$ and the cost $\max_{t \in T} C_{E_I}(t)$ is the *second-stage* cost. The objective is to find $E_I$ minimizing the two-stage cost (with the second-stage cost inflated by $\lambda$).

**Notation** For a set $P \subseteq V$ of vertices let $\partial P$ denote the *boundary* of $P$ — the set of edges that have exactly one endpoint in $P$. We use $E_I^*$ to denote the optimal solution.

### 2.1 Mincuts laminarity

Let $G^*$ be $G$ with the capacities of edges in $E_I^*$ set to 0. For a terminal $t \in T$, let $Q_t^* \subseteq V \setminus r$ denote the $t$-side of the cut $\mathsf{m}_{E_I^*}(t)$ — the vertices reachable from $t$ after the edges $E_I^*$ and $\mathsf{m}_{E_I^*}(t)$ are removed (the asterisk emphasizes that $Q^*$ is the $t$-side of the mincut in $G^*$, not in the original $G$). It is known (e.g., can be seen from the Gomory-Hu tree [11, Section 8.6])

that these $t$-sides do not properly intersect — $\forall u, v \in T$ either $Q^*_u \cap Q^*_v = \emptyset$ or $Q^*_u \subseteq Q^*_v$ or $Q^*_v \subseteq Q^*_u$. In other words, for any subset $S \subseteq T$ of terminals the $t$-sides of the terminals in $S$ form a laminar family $\mathcal{F}^*_S = \{Q^*_t : t \in S\}$ of sets.

Let $F^*_S \subseteq \mathcal{F}^*_S$ be the basic (inclusion-maximal) sets in the family $\mathcal{F}^*_S$; assume that all sets in $F^*_S$ are unique (note that in principle we could have $Q^*_u = Q^*_v = Q^*$ for $u, v \in S, u \neq v$, with $Q^*$ not being a proper subset of any other set in $\mathcal{F}^*_S$ — in this case only one of $Q^*_u, Q^*_v$ is included in $F^*_S$). Call the terminals $B^*_S = \{b \in S : Q^*_b \in F^*_S\}$ the *basic* terminals of $S$.

## 2.2 Thresholded $\alpha$-approximation

The thresholded covering paradigm applied to the robust mincut problem works as follows: Imagine that we know the cost $C^*_{II} = \max_{t \in T} C_{E^*_I}(t)$ that the optimum pays at the second stage. To obtain an $\alpha$-approximate solution, cut out the set $U = \{t \in T : C(t) > \alpha C^*_{II}\}$ of "expensive" terminals in the first stage. That is, the output of the algorithm is $\mathsf{m}(U)$.

Assuming the terminals in $T = (t_1, \ldots, t_{|T|})$ are ordered in non-increasing order of mincut ($C(t_i) \geq C(t_{i+1})$), for any $C^*_{II}$ we have $U = \{t_1, t_2, \ldots, t_j\}$ for some $j = j(C^*_{II}) \in \{0, 1, 2, \ldots, |T|\}$. Hence there are only $|T|+1$ different possible sets $U$ for all possible $C^*_{II} \geq 0$ — so try all the possibilities and choose the best; this way the non-determinism in $C$ is reduced just to the non-determinism w.r.t. $j$.

For the approximation ratio analysis assume that the algorithm is run with the "right" guess of $U$ corresponding to the right choice of $C^*_{II}$. By definition of $U$, the thresholded algorithm pays at most $\alpha C^*_{II}$ in the second stage. The tricky part is to bound the cost, $C(U)$, of the first stage.

**The analysis of [4]** Golovin, Goyal and Ravi [4] use the following estimate of the first-stage cost of the thresholded $\alpha$-approximation:

$$C(U) \quad \leq \quad C\Big(\bigcup_{b \in B^*_U} \partial Q^*_b\Big) \quad \leq \tag{1}$$

$$\leq \quad C(E^*_I) + \frac{1}{\alpha - 1} \sum_{b \in B^*_U} C(\partial Q^*_b \cap E^*_I)$$

To bound the last sum, [4] cleverly use the fact that due to pairwise-disjointness of the basic sets $F^*_U$, every edge $e \in E^*_I$ appears at most twice in the sum; thus $C(U) \leq (1 + \frac{2}{\alpha-1})C(E^*_I)$, and the overall approximation ratio of the thresholded $\alpha$-approximation algorithm is $\max(1 + \frac{2}{\alpha-1}, \alpha)$, minimized by $\alpha = 1 + \sqrt{2}$ — the final approximation ratio of [4].

### Using $\alpha = 2$ is enough

Our algorithm is just the thresholded 2-approximation, i.e., the output of our algorithm is the minimum $r$-$U$ cut where $U = \{t \in T : C(t) > 2C^*_{II}\}$; as usual, for the analysis we assume that $C^*_{II}$ (or, equivalently, $U$) was guessed correctly. The second-stage cost of our solution is at most $2C^*_{II}$. In what follows we prove the bound $C(U) \leq 2C(E^*_I)$ on the fist-stage cost of our algorithm.

To show $C(U) \leq 2C(E^*_I)$, instead of a correct but too generous bound (1) of [4] on $C(U)$, we use a tighter estimate

$$C(U) \quad \leq \quad C\left(\partial \bigcup_{b \in B^*_U} Q^*_b\right)$$

The correctness of the estimate follows from the same argument as in [4]: every terminal of $U$ belongs to at least one of the sets $Q^*_b$, and none of the sets $Q^*_b$ contains $r$; thus the boundary of the union is an $r$-$U$ cut.

To prove

$$C\left(\partial \bigcup_{b \in B^*_U} Q^*_b\right) \quad \leq \quad 2C(E^*_I)$$

we argue that

$$C\left(\partial \bigcup_{b \in B^*_U} Q^*_b\right) \quad \leq \quad 2C\left(\Delta^*\left(\bigcup_{b \in B^*_U} Q^*_b\right)\right) \tag{2}$$

where $\Delta^*(P) = ((P \times P) \cap E^*_I) \cup (\partial P \cap E^*_I)$ denotes the edges from $E^*_I$ that have *at least one* endpoint in a set $P \subseteq V$ of vertices; clearly, the right-hand-side of (2) is at most $2C(E^*_I)$.

Number the terminals in $B^*_U$ arbitrarily: $B^*_U = (b_1, b_2, \dots)$. For $k = 0, 1, \dots, |B^*_U|$ define $B^*_k = Q^*_{b_1} \cup Q^*_{b_2} \cup \dots \cup Q^*_{b_k}$. The inequality (2) follows from the next lemma:

**Lemma 2.1.** $\forall k = 0, 1, \dots, |B^*_U|, \quad C\left(\partial B^*_k\right) \leq 2C\left(\Delta^*\left(B^*_k\right)\right)$

*Proof.* By induction on $k$. The base is trivial: $0 = C(\partial\emptyset) \leq 2C(\Delta^*(\emptyset)) = 0$.

Let $X^* = \partial(B^*_{k-1}, Q^*_{b_k}) \cap E^*_I, X = \partial(B^*_{k-1}, Q^*_{b_k}) \setminus E^*_I, Y^* = (\partial Q^*_{b_k} \setminus (X^* \cup X)) \cap E^*_I, Y = (\partial Q^*_{b_k} \setminus (X^* \cup X)) \setminus E^*_I$ (Fig. 1).

Because $b_k$ belongs to the set of high-cost terminals $U$, the optimal solution $E^*_I$ must "help" $b_k$ by at least half (using the terminology from [7], $b_k$ is "low"):

$$C(X^*) + C(Y^*) \geq C(X) + C(Y) \tag{3}$$

Indeed, since $\partial Q^*_{b_k}$ is an $r$-$b_k$ cut, $C(b_k) \leq C(\partial Q^*_{b_k}) = C(X^*) + C(Y^*) + C(X) + C(Y)$, and since $b_k \in U$, $C(b_k) \geq 2C^*_{II} \geq 2C_{E^*_I}(b_k) = 2(C(X) + C(Y))$, from where (3) follows.

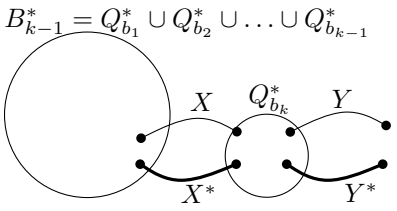


Figure 1: $E^*_I$ is bold. $B^*_k = B^*_{k-1} \cup Q^*_{b_k}$.

Using $C(X) \geq 0$ we rewrite (3) as

$$C(Y^*) \geq \frac{C(Y^*)}{2} + \frac{C(Y) - C(X^*) + C(X)}{2} \geq \\ \geq \frac{C(Y^*) + C(Y) - C(X^*) - C(X)}{2} \tag{4}$$

We have (see Fig 1):

$$C(\Delta^*(B_k)) \geq C(\Delta^*(B^*_{k-1})) + C(Y^*) \tag{5}$$

$$C(\partial B^*_k) = C(\partial B^*_{k-1}) + C(Y^*) + C(Y) - C(X^*) - C(X) \tag{6}$$

By the inductive hypothesis,

$$C(\Delta^*(B_{k-1})) \geq \frac{C(\partial B^*_{k-1})}{2} \tag{7}$$

Putting (4), (5), (6), (7) together we obtain

$$C(\Delta^*(B_k)) \geq \frac{C(\partial B^*_{k-1})}{2} + \frac{C(Y^*) + C(Y) - C(X^*) - C(X)}{2} = \frac{C(\partial B^*_k)}{2}$$

□

Overall, we have that the first-stage cost of our solution is at most $2C(E^*_I)$, and the second-stage cost is at most $2C^*_{II}$:

**Theorem 2.2.** *There is a polynomial-time algorithm which gives a 2-approximation for the robust mincut problem.*

# 3 Demand-robust shortest path

The input to the demand-robust shortest path is the same as to the demand-robust mincut problem: graph $G = (V, E, w)$ with $w$ representing the *lengths* of edges, root vertex $r \in V$, and a set $T \subseteq V \setminus r$ of terminals. A solution is a set $E_I \subseteq E$ of edges. The cost of the solution is

$$w(E_I) + \lambda \cdot \max_{t \in T} w(\mathsf{SP}_{E_I}(t))$$

where $\mathsf{SP}_{E_I}(t)$ is the shortest $r$-$t$ path in $G$ with weights of edges in $E_I$ set to 0. The objective is to find $E_I$ minimizing the two-stage cost.

### 3.1 Algorithm

Our solution is a Steiner tree on a subset $S \subseteq T$ of $\lambda$ terminals. The set $S$ is built incrementally, starting from $r$, and repeatedly adding a farthest (with ties broken arbitrarily) terminal, until gathering $\lambda$ of them:

1 $S \leftarrow \{r\}$
2 **while** $|S| \leq \min(\lambda, |T|)$
3 **do** $S \leftarrow S \cup \arg\max_{u \in T \setminus S} \mathsf{SP}(u, S)$ $\triangleright$ Add farthest terminal
4 **return** $E_I \leftarrow$ approximate Steiner tree on $S$

### 3.2 Approximation ratio

We now analyze the approximation guarantee of the algorithm. Let $f$ be the distance from $S$ to the terminal added in the last iteration of the **while** loop (line 3). Because $|S|$ was growing from iteration to iteration, we have that at *any* iteration the distance from $S$ to the farthest terminal was at least $f$. Thus, the distance between any two vertices in $S$ is at least $f$. Hence, $\mathsf{St}(S) \geq |S|f/2$ where $\mathsf{St}(S)$ is the weight of the minimum Steiner tree on $S$ (to see this, go twice around the tree — you traveled $2\mathsf{St}(S)$, spending at least $f$ traveling between any two vertices in $S$). That is,

$$f \leq \frac{2\mathsf{St}(S)}{|S|}$$

Because we always add farthest terminal to $S$, at the completion of the algorithm the distance from any terminal to $S$ is at most $f$. Hence,

$$w(E_{II}) \leq f$$

where $E_{II}$ are the edges that we buy at the second stage.

Let $E_I^*$ be the optimal solution. Let $E_{II}^*(t)$ be the edges that the optimal solution buys at the second stage if the demand is $t \in T$. Let $E_{II}^*$ be the edges that the optimum buys in the worst case: $w(E_{II}^*) = \max_t w(E_{II}^*(t))$. Then $E_I^* \cup_{t \in S \setminus r} E_{II}^*(t)$ is a connected graph that spans $S$. Hence

$$w(E_I^*) + \sum_{t \in S \setminus r} w(E_{II}^*(t)) \geq \mathsf{St}(S)$$

and

$$w(E_I^*) + (|S| - 1)w(E_{II}^*) \geq \mathsf{St}(S)$$

We consider the cases $\lambda \geq |T|$ and $\lambda < |T|$ separately:
If $\lambda \geq |T|$, then $S = T \cup r$, and

$$\begin{aligned} \mathsf{opt} = w(E_I^*) + \lambda w(E_{II}^*) \quad &\geq \quad w(E_I^*) + |T| w(E_{II}^*) \quad = \\ = \quad w(E_I^*) + (|S| - 1)w(E_{II}^*) \quad &\geq \quad \mathsf{St}(S) \quad \geq \quad w(E_I)/\gamma \quad = \quad \mathsf{apx}/\gamma \end{aligned}$$

where $\mathsf{opt}$ is the optimal cost, $\mathsf{apx}$ is what we pay, and $\gamma$ is the approximation factor for the Steiner tree. That is, $\mathsf{apx} \le \gamma\,\mathsf{opt}$.

If $\lambda < |T|$, then

$$\begin{aligned}\mathsf{apx} &= w(E_I) + \lambda w(E_{II}) \le \gamma \mathsf{St}(S) + \lambda f \le \\ &\le \left(\gamma + \frac{2\lambda}{|S|}\right)\mathsf{St}(S) \le \left(\gamma + \frac{2\lambda}{|S|}\right)(w(E_I^*) + (|S|-1)w(E_{II}^*)) \le \\ &\le (\gamma+2)(w(E_I^*) + \lambda w(E_{II}^*)) = (\gamma+2)\,\mathsf{opt}\end{aligned}$$

because $\lambda \le |S| \le \lambda + 1$.

**Theorem 3.1.** *If for some class of graphs there is a $\gamma$-approximation for Steiner tree, then for that class of graphs there is a $(\gamma+2)$-approximation for robust shortest path.*

For general graphs, the best $\gamma = 1.39$ is due to Byrka et al. [1]:

**Corollary 3.2.** *There is a polynomial-time algorithm which gives a 3.39-approximation for the robust shortest path problem.*

### 3.3 Running time

As far as the efficiency of approximating the robust shortest path is concerned, the best bound that can be given on the running time of the algorithm of [4] is $O(|T||V||E|)$. The (multiplicative) overhead of $O(T||V|)$ is due to guessing $|T||V|$ possible values for the second-stage cost of the optimal solution. Then for each guess the algorithm of [4] builds an approximate Steiner tree, which must take $\Omega(|E|)$ time.

Because our algorithm avoids the guessing, we can achieve the running time of $O(\min(\lambda, |T|)(|E| + |V|\log|V|))$ at the expense of increasing the approximation ratio to 4. For that, in line 4 we use the $O(|E| + |V|\log|V|)$-time 2-approximation algorithm of Mehlhorn [12] or Floren [3]. Then our algorithm's running time is dominated by finding the farthest terminals in line 3.

**Corollary 3.3.** *There is an $O(\min(\lambda, |T|)(|E| + |V|\log|V|))$-time algorithm which gives a 4-approximation for the robust shortest path problem.*

## References

[1] J. Byrka, F. Grandoni, T. Rothvoß, and L. Sanità. An improved lp-based approximation for steiner tree. In *STOC '10: Proceedings of the 42nd ACM symposium on Theory of computing*, pages 583–592, New York, NY, USA, 2010. ACM.

[2] K. Dhamdhere, V. Goyal, R. Ravi, and M. Singh. How to pay, come what may: Approximation algorithms for demand-robust covering problems. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 367–378. IEEE Computer Society, 2005.

[3] R. Floren. A note on "a faster approximation algorithm for the steiner problem in graphs". *Information Processing Letters*, 38(4):177–178, 1991.

[4] D. Golovin, V. Goyal, and R. Ravi. Pay today for a rainy day: Improved approximation algorithms for demand-robust min-cut and shortest path problems. In B. Durand and W. Thomas, editors, *STACS 2006, 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006, Proceedings*, volume 3884 of *Lecture Notes in Computer Science*, pages 206–217. Springer, 2006.

[5] F. Grandoni, A. Gupta, S. Leonardi, P. Miettinen, P. Sankowski, and M. Singh. Set covering with our eyes closed. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 347–356. IEEE Computer Society, 2008.

[6] A. Gupta, V. Nagarajan, and R. Ravi. Thresholded covering algorithms for robust and max-min optimization. *CoRR*, abs/0912.1045, 2009.

[7] A. Gupta, V. Nagarajan, and R. Ravi. Thresholded covering algorithms for robust and max-min optimization. In S. Abramsky, C. Gavoille, C. Kirchner, F. M. auf der Heide, and P. G. Spirakis, editors, *Automata, Languages and Programming, 37th International Colloquium, ICALP 2010, Bordeaux, France, July 6-10, 2010, Proceedings, Part I*, volume 6198 of *Lecture Notes in Computer Science*, pages 262–274. Springer, 2010.

[8] A. Gupta, M. Pál, R. Ravi, and A. Sinha. Boosted sampling: approximation algorithms for stochastic optimization. In L. Babai, editor, *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 417–426. ACM, 2004.

[9] A. Gupta, R. Ravi, and A. Sinha. An edge in time saves nine: LP rounding approximation algorithms for stochastic network design. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 218–227. IEEE Computer Society, 2004.

[10] L. Jia, G. Lin, G. Noubir, R. Rajaraman, and R. Sundaram. Universal approximations for tsp, steiner tree, and set cover. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 386–395, New York, NY, USA, 2005. ACM.

[11] B. H. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics, 21)*. Springer, 2008.

[12] K. Mehlhorn. A faster approximation algorithm for the steiner problem in graphs. *Information Processing Letters*, 27(3):125–128, 1988.